DOI 10.37882/2223-2966.2024.05.20

АВТОМАТИЗАЦИЯ СБОРКИ ОБРАЗОВ И ТЕСТИРОВАНИЕ С ПОМОЩЬЮ JENKINS НА ЯНДЕКС.ОБЛАКЕ

AUTOMATING IMAGE BUILDS AND TESTING WITH JENKINS ON YANDEX.CLOUD

Liu Yuanzhi
V. Borisov

Summary. DevOps work on Continuous Integration/Continuous Deployment depends on the speed of deployment and on managing the tasks associated with this process. Jenkins, GitHub can help with this. Deployment automation tools will reduce the time it takes to build and configure virtual machine images and make it easier to scale your infrastructure. Automate image building and testing to automate software deployment and reduce the consumption of computer and time resources. Create a tool to automatically build images on Linux systems. To achieve this goal, standard Linux utilities are used. The traditional approach to image building requires knowledge of computer networks. We recommend automatic image builds on linux systems. Based on this situation, the software suite should contain the following components: HTTP server, message broker, and hardware device connections, web interface or special programs. One solution is a software suite: Jenkins, Packer, jq. At the moment, the programs are installed manually on different virtual or physical machines and require knowledge of the host machines operating systems device, as well as knowledge of computer networks and some administration skills. As a result, the following will be created: 1) Cloud network. 2) Sub-networks in all availability zones. 3) VMs from images created with Packer. VMs with nginx will get public IP addresses. All VMs will be connected to subnets. Automating image builds with Jenkins and Packer has been covered in detail. Existing frameworks for image build automation are diverse, each has its own features and drawbacks. The work with Packer was considered on its own example, as a result of which the ease of operation, usability and quick startup were evaluated.

Keywords: DevOps, Jenkins, GitHub, deployment automation, image build automation.

Лю Юаньчжи
Уральский Федеральный Университет, Екатеринбург
liuliu18845790183@163.com
Борисов Василий Ильич
кандидат технических наук, доцент,
Уральский Федеральный Университет, Екатеринбург
v.i.borisov@urfu.ru

Аннотация. Работа DevOps по непрерывной интеграции и развертыванию приложений (Continuous Integration/Continuous Deployment) зависит от скорости развёртывания и от управления задачами, связанными с этим процессом. В этом помогут Jenkins, GitHub. Средства автоматизации развертывания сократят время на сборку и настройку образов виртуальных машин и облегчат масштабирование инфраструктуры [1]. Автоматизировать сборку образов и тестирование для автоматизации развертывания программного обеспечения и уменьшения потребления компьютерных и временных ресурсов. Создать инструмент для автоматической сборки образов на Linux системы. Для достижения данной цели используются стандартные утилиты Linux. Традиционный подход к сборке образов требует знания компьютерных сетей. Рекомендуем автоматические сборки образов на системах linux. Исходя из этой ситуации, набор программного обеспечения должен содержать следующие компоненты: HTTP-сервер, брокер сообщений, и соединения аппаратных устройств, веб-интерфейс или специальные программы. Одно из решений — набор программного обеспечения: Jenkins, Packer, jq. На данный момент программы устанавливаются вручную на разных виртуальных или физических машинах и требуют знаний устройства операционных систем хост-машин, а также знаний компьютерных сетей и некоторых навыков администрирования [2]. В результате будут созданы: 1) Облачная сеть. 2) Подсети во всех зонах доступности. 3) ВМ из образов, созданных с помощью Packer. ВМ с nginx получат публичные IP-адреса. Все ВМ будут подключены к подсетям [3]. Автоматизация сборки образов с помощью Jenkins и Packer было рассмотрено подробно. Существующие фреймворки для автоматизаций сборки образов разнообразны, каждый имеет свои особенности и недостатки. На собственном примере была рассмотрена работа с Packer, в результате чего оценены легкость в работе, удобство использования и быстрота запуска.

Ключевые слова: DevOps, Jenkins, GitHub, автоматизации развертывания, автоматизация сборки образов.

Введение

Актуальность темы. Существует мнение, что тесты не нужны, если у нас достаточная сборка образов. Но от следующей метафоры никуда не деться: представим, что мы собираем машину. У нас есть четыре хорошо протестированных колеса, протестированная рама вместе с седлом. То есть мы имеем хороший набор тестов. А машина-то в итоге поедет? Чтобы это проверить, нужно нанимать ручных тестировщиков, которые перед каждым релизом должны убедиться, что безупречные детали корректно взаимодействуют друг с другом, и машина будет ездить и доставлять пользователю удовольствие [4].

В данном проекте нужно автоматизация сборки образов на Yandex.Cloud. Yandex.Cloud предоставляет частным и корпоративным пользователям инфраструктуру и вычислительные ресурсы в формате as a service [5].

Большое влияние уделяется заключительному этапу производства в технологии машиностроения, а именно

сборке изделия. Важность сборки объясняется тем, что ее результаты в значительной мере определяют производственно-техническое и эксплуатационное качество изделия. Сложность выработки алгоритма выбора оптимального решения поставленной задачи заключается в ее комплексный характере и требует системного подхода с учетом влияния взаимосвязанных конструкторско-технологических факторов. Поэтому разработка научно обоснованных технологических процессов, а также методик, рекомендаций и критериев эффективности, является актуальной задачей [6].

При командной распределенной разработке больших проектов по созданию программного обеспечения в настоящее время широко применяются автоматизации сборки образов. Данная технология использует виртуализацию и средства управления конфигурациями виртуальных машин для применения необходимых параметров и установки требуемых компонентов, автоматизируя процесс синхронизации, настройки и запуска рабочего окружения [7].

Все программы устанавливаются вручную на разных виртуальных или физических машинах и требуют знаний устройства операционных систем хост-машин, а также знаний компьютерных сетей и некоторых навыков администрирования. Достоинства автоматизированной сборки образов по сравнению с ручной сборкой образов:

- Автоматизированные методы компиляции способны преобразовывать исходный код в исполняемые программные приложения, уменьшая количество ошибок в процессе компиляции, выполняемой человеком.
- Технология автоматизированного тестирования позволяет автоматически выполнять различные тестовые сценарии для проверки стабильности и корректности работы программного обеспечения в различных средах и условиях.

Технологии автоматизированного развертывания упрощают, ускоряют и повышают надежность процесса развертывания сборки образов.

1. Автоматизация для Яндекс.Облаков

В данном проекте можно подключать к виртуальным машинам диски с образами на базе ОС Linux, доступные в Marketplace. Каждый диск автоматически реплицируется внутри своей зоны доступности, что обеспечивает надежное хранение данных. Также, для удобного переноса данных с одного диска на другой, Compute Cloud поддерживает снимки дисков [8]. Автоматизация развёртывания — одна из самых сложных тем в сфере разработки ПО. По различным наблюдениям, немногие в сообществе автоматизируют развёртывание, а те, кто это делают, не всегда получают реальную пользу. Кроме того, подступиться к теме не так-то просто: материалы в основном разрозненные, не всегда актуальны для нужной платформы, а в чем-то и противоречивы. В общем, чтобы начать нормально автоматизировать развёртывание, необходимо очень много искать и разбираться.

Преимущества автоматизированного развертывания:

- Если систему необходимо развернуть распределенным образом, то экономия времени будет экспоненциальной. Пока экономия времени велика, можно применять решение по автоматизированному развертыванию. Это первая точка автоматического развертывания, экономия времени.
- Второе преимущество автоматизированного развертывания заключается в том, что процесс развертывания осуществляется в соответствии со строгими спецификациями. На самом деле, многие компании разработали набор спецификаций процессов и систем, которые подходят для их собственной компании в области проектирования, разработки, тестирования и эксплуатации и технического обслуживания. Но в реальности эти процессы и спецификации постепенно размываются со временем, а в некоторых компаниях весь проект находится в хаосе, о котором они даже не подозревают, когда его создают. Машина, однако, отличается от других и всегда будет следовать разработанной вами программе в точности.
- Третьим преимуществом автоматизированного развертывания является 100% точность ввода команд. Это вполне объяснимо, поскольку люди постоянно устают, расстраиваются, находятся в разном настроении и состоянии, и даже если они в полном порядке, они могут допустить опечатку при наборе команды. Если вы допустите опечатку во время части процесса развертывания «изменить профиль», система не сообщит об ошибке, и только после запуска службы и обнаружения различных проблем вы вернетесь и проверите предыдущие шаги, что займет много времени. Для автоматизированного развертывания это является отличным преимуществом.
- Снизить нагрузку на тестировщиков и повысить эффективность.

Но есть и проблемы:

- Нужно время на внедрение, написание и поддержку;
- При некорректном внедрении практики могут принести больше вреда, чем пользы.

После публикации артефактов сборки в ходе постоянной интеграции следующим шагом становится их развертывание в тестовой среде для автоматического

тестирования: интеграционных тестов, сквозного тестирования, тестов производительности и безопасности. Затем выполняется ручное исследовательское тестирование и сбор обратной связи.

На последнем этапе изменения выпускаются в производственную среду либо в автоматизированном режиме (непрерывное развертывание), либо с помощью запускаемого вручную скрипта (непрерывная доставка).

Рекомендуется использовать во всех окружениях одни и те же артефакты сборки, каждый раз извлекая их из репозитория артефактов, чтобы процесс развертывания везде был максимально похожим.

В этом случае вы протестируете процесс многократно для каждой сборки еще до выпуска в продакшн и будете уверены в ее качестве. Если ваша организация только знакомится с CI/CD и DevOps, договориться о едином процессе развертывания может быть сложно: командам нужно будет выстроить совместную работу для достижения общей цели.

Автоматизация процесса развертывания — необходимое условие для частого выпуска обновлений. Без нее вам придется вручную обновлять тестовые среды и развертывать новые сборки каждый раз, когда нужно запустить режим автоматического тестирования этих сборок. В результате вы будете позже получать обратную связь, и доставка изменений пользователям займет больше времени [9].

2. Фреймворки для автоматизации сборки образов

Рассмотрим существующие фреймворки для автоматизации сборки образов Яндекс. Для каждого типа сначала приведена сравнительная таблица возможностей инструментов, которые к нему относятся. В таблице собрана самая актуальная и достоверная информация о каждом инструменте (см. таблицу 1) [10].

Таблица 1.

Особенности отдельных фреймворков

Инструмент	Языки	Доступ к исходникам	Реальные устройства
Jenkins	Javascript	–	+
Packer	Javascript	+	+
GitHub	Shell	+	+
Terraform	Javascript	–	+

Jenkins

Jenkins позволяет автоматизировать часть процесса разработки программного обеспечения, в котором не обязательно участие человека, обеспечивая функции непрерывной интеграции. Работает в сервлет-контейнере, например, Apache Tomcat. Поддерживает инструменты системы управления версиями, включая AccuRev, CVS, Subversion, Git, Mercurial, Perforce, Clearcase и RTC. Может собирать проекты с использованием Apache Ant и Apache Maven, а также выполнять произвольные сценарии оболочки и пакетные файлы Windows. Сборка может быть запущена разными способами, например, по событию фиксации изменений в системе управления версиями, по расписанию, по запросу на определённый URL, после завершения другой сборки в очереди. Jenkins также может автоматизировать развертывание Яндекс-сайтов [11].

Packer

Packer — это инструмент с открытым исходным кодом для создания идентичных машинных образов для нескольких платформ из одной исходной конфигурации. Packer имеет небольшой вес, работает на всех основных операционных системах и обладает высокой производительностью, создавая образы машин для нескольких платформ параллельно. Образ машины — это единый статический блок, содержащий предварительно сконфигурированную операционную систему и установленное программное обеспечение, который используется для быстрого создания новых работающих машин. Форматы машинных образов меняются для каждой платформы. Некоторые примеры включают AMI для EC2, файлы VMDK/VMX для VMware, экспорт OVF для VirtualBox и т.д. Более простое автоматическое развертывание образов Яндекс с помощью Packer [12].

GitHub

GitHub — крупнейший веб-сервис для хостинга IT-проектов и их совместной разработки.

Веб-сервис основан на системе контроля версий Git и разработан на Ruby on Rails и Erlang компанией GitHub, Inc (ранее Logical Awesome). Сервис бесплатен для проектов с открытым исходным кодом и (с 2019 года) небольших частных проектов, предоставляя им все возможности (включая SSL), а для крупных корпоративных проектов предлагаются различные платные тарифные планы [13]. Чтобы Jenkins мог выполнять сборки образов, следует указать авторизационные данные для Yandex Cloud и создать задачу на получение изменений из репозитория GitHub. Авторизационные данные будут использоваться в переменных, находящихся в конфигурационных файлах Packer.

В настройках репозитория GitHub включите webhook для запуска сборки в Jenkins и добавьте публичный SSH-ключ для авторизации.

Сборка образа в Jenkins запускается автоматически после выполнения команды push в ветке master GitHub-репозитория.

Terraform

HashiCorp Terraform — это инструмент инфраструктуры как кода, позволяющий определять облачные и локальные ресурсы в человекочитаемых конфигурационных файлах, которые можно редактировать, повторно использовать и совместно использовать. Затем вы можете использовать последовательный рабочий процесс для обеспечения и управления всей вашей инфраструктурой на протяжении всего ее жизненного цикла. Terraform может управлять низкоуровневыми компонентами, такими как вычислительные ресурсы, ресурсы хранения и сетевые ресурсы, а также высокоуровневыми компонентами, такими как записи DNS и функции SaaS.

Terraform позволяет быстро создать облачную инфраструктуру в Yandex Cloud и управлять ею с помощью файлов конфигураций. В файлах конфигураций хранится описание инфраструктуры на языке HCL (HashiCorp Configuration Language). Terraform и его провайдеры распространяются под лицензией Mozilla Public License.

3. Написание сборки образов и тестирования

Перед работой нужно зарегистрироваться в Yandex Cloud и создать платежный аккаунт:

1. Перейдите в консоль управления, затем войдите в Yandex Cloud или зарегистрируйтесь, если вы еще не зарегистрированы.
2. На странице биллинга убедитесь, что у вас подключен платежный аккаунт, и он находится в статусе ACTIVE или TRIAL_ACTIVE. Если платежного аккаунта нет, создайте его.

Если у вас есть активный платежный аккаунт, вы можете создать или выбрать каталог, в котором будет работать ваша инфраструктура, на странице облака.

Jenkins будет получать изменения в конфигурациях образов ВМ из GitHub, а затем с помощью Packer создавать образы в облаке.

Packer позволяет создавать образы дисков виртуальных машин с заданными в конфигурационном файле параметрами.

Чтобы Jenkins мог выполнять сборки образов, следует указать авторизационные данные для Yandex Cloud и создать задачу на получение изменений из репозитория GitHub. Авторизационные данные будут использоваться в переменных, находящихся в конфигурационных файлах Packer.

В настройках репозитория GitHub включите webhook для запуска сборки в Jenkins и добавьте публичный SSH-ключ для авторизации.

Сборка образа в Jenkins запускается автоматически после выполнения команды push в ветке master GitHub-репозитория (см. рисунок 1).

После того как образы будут созданы, их можно использовать для создания ВМ. Создайте тестовую инфраструктуру с помощью Terraform (см. рисунок 2).

После этого будут созданы:

1. Облачная сеть (см. рисунок 3).
2. Подсети во всех зонах доступности (см. рисунок 4).
3. ВМ из образов, созданных с помощью Packer. ВМ с nginx получат публичные IP-адреса. Все ВМ будут подключены к подсетям (см. рисунок 5).

4. Выводы

Подводя итоги проекта, можно утверждать, что автоматизация сборки образов Яндекс было рассмотрено подробно. Существующие фреймворки для автоматизаций сборки образов разнообразны, каждый имеет свои особенности и недостатки. На собственном примере была рассмотрена работа с Packer, в результате чего оценены легкость в работе, удобство использования и быстрота запуска. Оба созданных автоматизации сборки образов в точности выдали требуемый результат. Таким образом, результаты проекта можно описать следующими пунктами:

1. Изучены существующие фреймворки для автоматизации сборки образов Яндекс;
2. Внедрена автоматизация сборки образов в Яндекс Облаке;

Образы

Фильтр по имени | Все статусы

Имя	Описание	Размер	Статус	Дата создания	Оптимизировать для развертывания	Идентификатор	Метки
debian-11-nginx-2023-12-20t16-20-33z	My very custom nginx proxy	10 ГБ	Ready	20.12.2023, в 21:22	Нет	fd8dkg4pu22tc4m1au7h	—
debian-11-base-2023-12-20t16-17-39z	Yet another debian build 2	10 ГБ	Ready	20.12.2023, в 21:19	Нет	fd8heh19v8vnbjmgbso8	—
debian-11-django-2023-12-20t16-20-33z	My very custom Django build	10 ГБ	Ready	20.12.2023, в 21:23	Нет	fd8sbphdq4nb35hvg1or	—

Рис. 1. Сборка образа

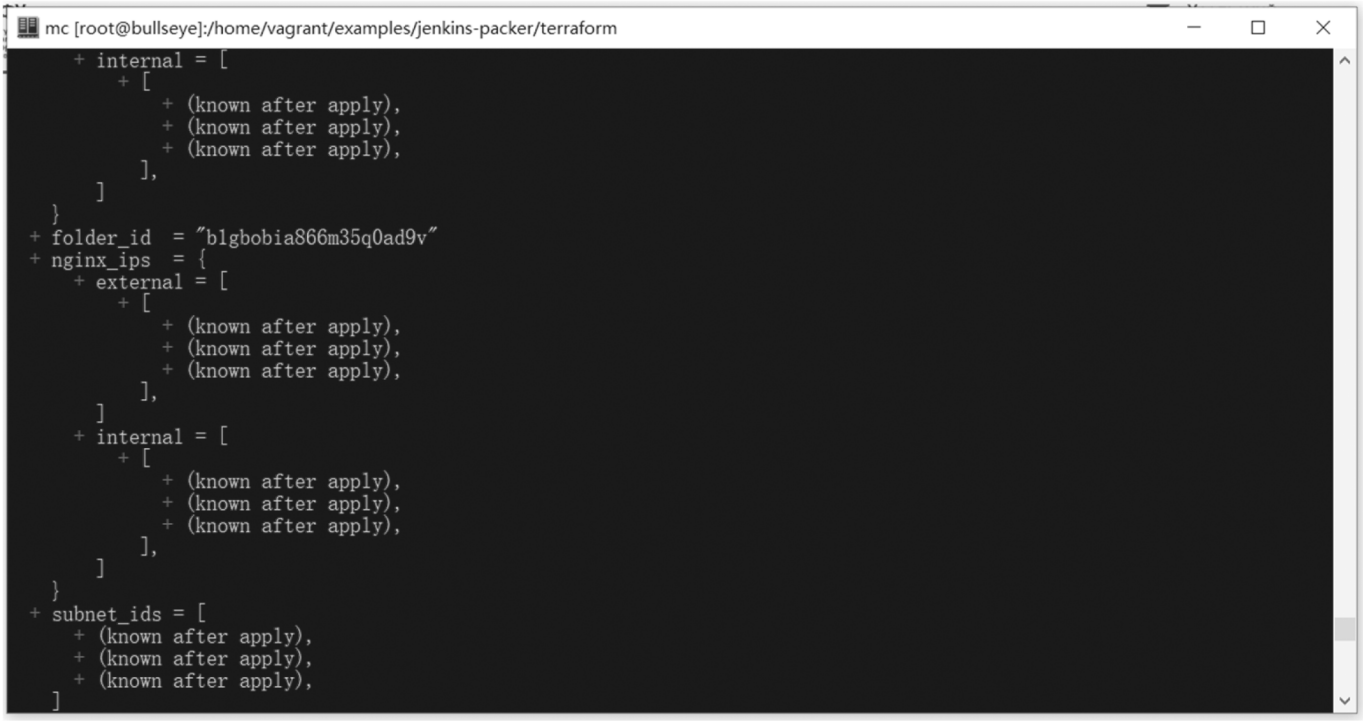

Рис. 2. Тестовая инфраструктура

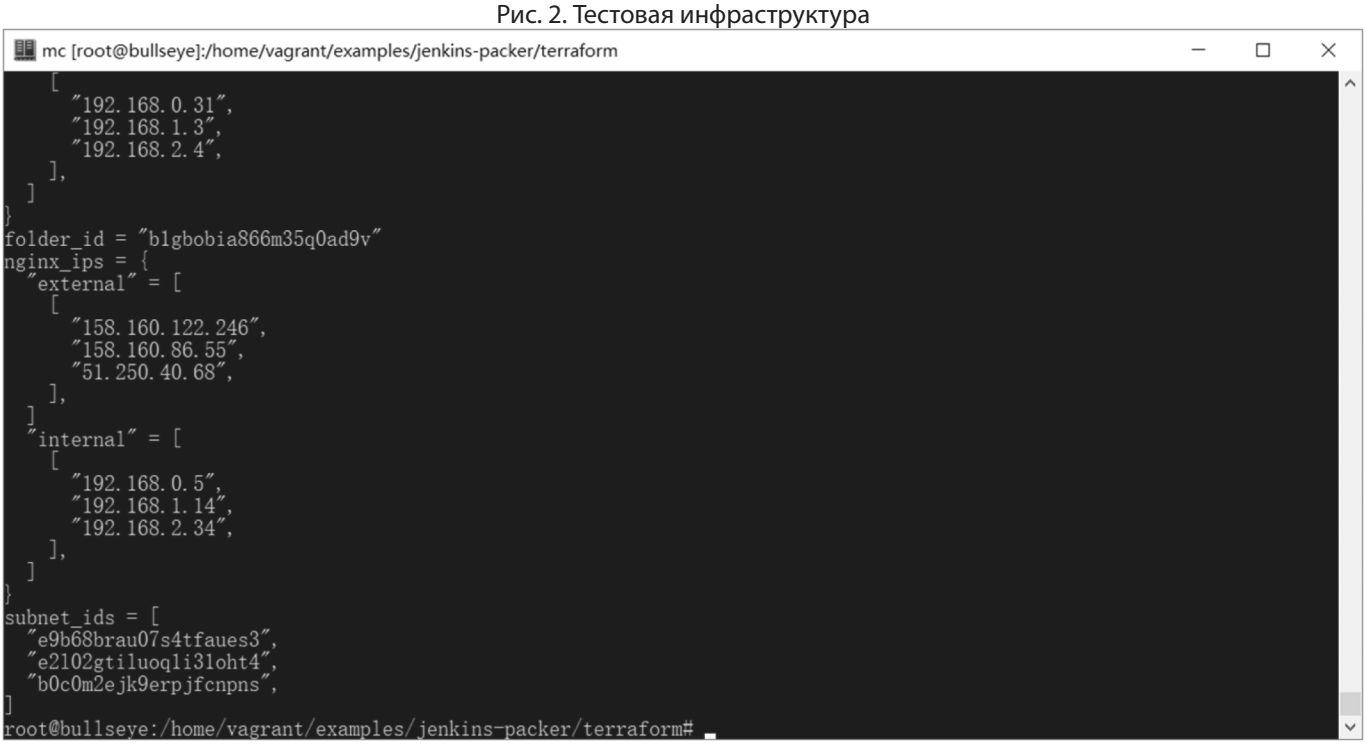

Рис. 3. Облачная сеть

Рис. 4. Подсети во всех зонах доступности

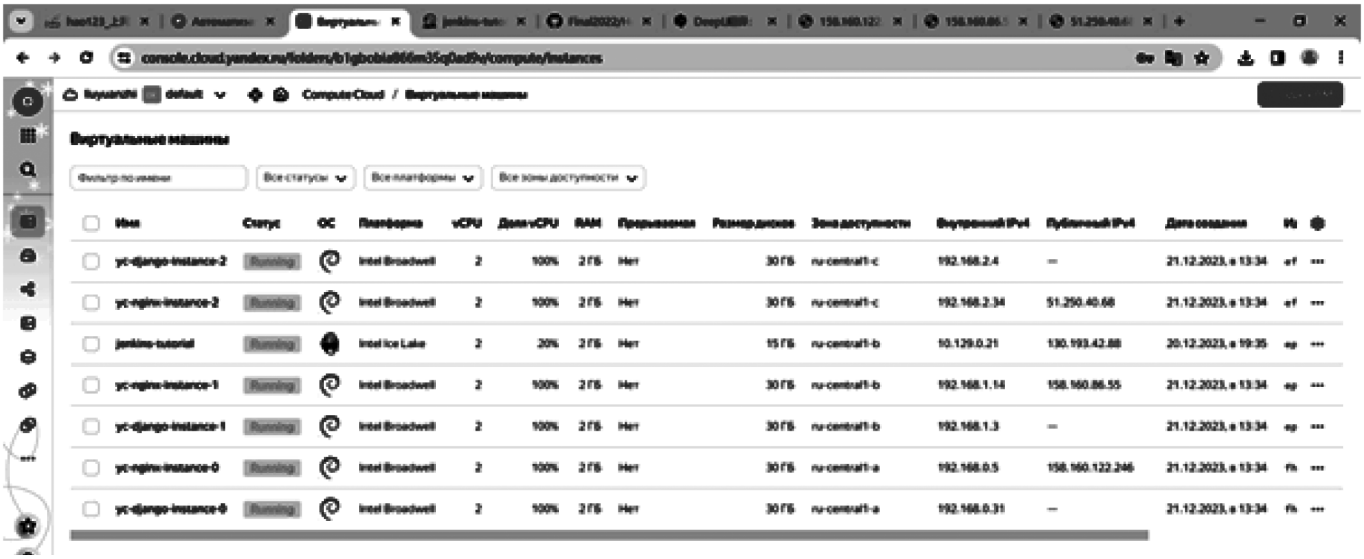

Рис. 5. ВМ из образов

3. Приобретены возможности автоматической сборки образов.

В ходе работы была спроектирована, программно реализована и исследована автоматизации сборки образов. Сборка образов обеспечивает надежную и качественную работу ее технических объектов и систем в едином стеке при различных типах воздействий и инцидентах, самостоятельно обучается и предоставляет интеллектуальную поддержку при принятии управленческих решений в технических системах в условиях неопределенности.

Объединение всех данных в единый стек является дополнительным достоинством перед существующими решениями [14].

Обобщая все вышесказанное о процессе разработки программного обеспечения, можно выделить следующие перспективы автоматизации данной деятельности:

1. унификация приемов и технологий;
2. упаковка всех этапов разработки программного обеспечения в единую сервисную модель автоматической генерации продукта по типу «черного ящика»;
3. объединение профессиональных ролей [15];

Необходимо автоматизировать процессы для автоматизации сборки образов с минимально возможным количеством уязвимостей там, где это может быть сделано, а также использовать комбинации различных инструментов и подходов для обеспечения качества программного обеспечения [16].

ЛИТЕРАТУРА

1. Автоматизация сборки образов и тестирование с помощью Jenkins на Яндекс.Облаке — https://events.yandex.ru/events/webinars/mar-28/ (дата обращения: 16.09.2023).
2. Буров В.С. Автоматизация развертывания системы «умный город» // Алтайский государственный университет. — 2019. — С. 134–135. eLIBRARY ID: 38226016.
3. Автоматизация сборки образов с помощью Jenkins и Packer — https://cloud.yandex.ru/docs/tutorials/infrastructure-management/jenkins (дата обращения: 16.09.2023).
4. Чимитов П.Е. Построение последова тельности сборки планер а самолета на основе образа изделия // Иркутский государственный технический университет. — 2009. — С. 218–222. eLIBRARY ID: 12808290.
5. Яндекс.Облако — https://ru.wikipedia.org/wiki/%D0%AF%D0%BD%D0%B4%D0%B5%D0%BA%D1%81.%D0%9E%D0%B1%D0%BB%D0%B0%D0%BA%D0%BE (дата обращения: 16.09.2023).
6. Борисов В.М., Лашков В.А., Борисов С.В. Критерии оценки оптимального варианта технологических процессов сборки в компрессоростроении // КНИТУ. — 2020. — С. 91–93. eLIBRARY ID: 43794529.
7. Колясников П.В., Силаков И.Н., Ильин Д.Ю., Гусев А.А., Никульчев Е.В. Повышение эффективности виртуального рабочего окружения распределенной разработки программ // Российская академия образования, Москва, Россия, МИРЭА — Российский технологический университет, Москва, Россия, Кубанский государственный университет, Краснодар, Россия. — 2019. — С. 72–80. eLIBRARY ID: 38468907.
8. Yandex Compute Cloud — https://cloud.yandex.ru/docs/compute/ (дата обращения: 16.09.2023).
9. Что такое автоматизация развертывания? — https://www.jetbrains.com/ru-ru/teamcity/ci-cd-guide/concepts/deployment-automation/ (дата обращения: 16.09.2023).

10. Программа для развертывания приложений на базе sel4 с помощью фреймворка camkes / Автономная некоммерческая организация высшего образования «Университет Иннополис». // Университет Иннополис. — 2022. eLIBRARY ID: 49199038.
11. Jenkins (программное обеспечение) — https://ru.wikipedia.org/wiki/Jenkins_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) (дата обращения: 16.09.2023).
12. What is Packer? — https://developer.hashicorp.com/packer/docs/intro (дата обращения: 16.09.2023).
13. GitHub — https://ru.wikipedia.org/wiki/GitHub (дата обращения: 16.09.2023).
14. Басыня Е.А. Программная реализация и исследование системы интеллектуально-адаптивного управления информационной инфраструктурой предприятия // Новосибирский государственный технический университет, Научно-исследовательский институт информационно-коммуникационных технологий, 630073, г. Новосибирск, пр-т К. Маркса, 20. — 2020. eLIBRARY ID: 43477195.
15. Вичугова А.А. Автоматизация процесса разработки программного обеспечения: методы и средства // Томский политехнический университет. — 2016. eLIBRARY ID: 26236252.
16. Тулеубаева А.А., Камолов А.Б., Рычков В.А. Современные методологии разработки безопасного программного обеспечения // НИЯУ МИФИ. — 2022. eLIBRARY ID: 47961926.

Журнал «Современная наука: актуальные проблемы теории и практики»